Deflategate
A data driven investigation

Lecture plan

- Deflategate - class discussion
- NFL analytics
- Analytics and Moneyball

- Structure of the internet
- RESTful API
- XML format
- Json format
- Algorithms for extracting data from XML and JSON

Class Discussion

1. What are the main facts?
2. What data would support the deflation theory?
3. What data would help refute the deflation theory?
4. How would we go about resourcing this data and what analytics would we apply?

Analytics in the NFL

Based on the presentation:

Going for Three: Predicting the Likelihood of Field Goal Success with Logistic Regression

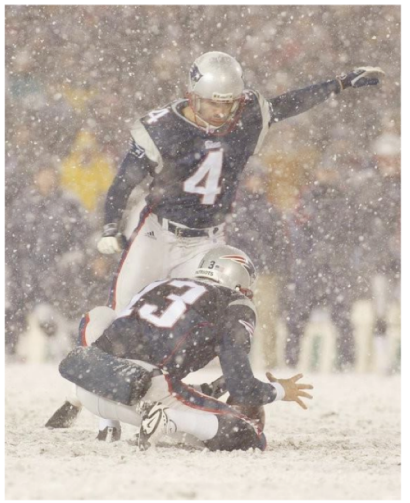
Torin Clark, Aaron Johnson, Alexander Stimpson
Massachusetts Institute of Technology

Buffalo Bills at Miami Dolphins
December 23, 2012 (Week 17)
Miami – 21, Buffalo – 3
3rd quarter, 4:22

45 yards	45 yards
Warm	Cold
Sunny	Snowy
Calm	Windy
Regular season	Postseason
No pressure	High pressure

All 45 yard field goals are not created equal.

Which of these factors really make a difference?



Oakland Raiders at New England Patriots
January 19, 2002 (Divisional Playoffs)
Oakland – 13, New England – 10
4th quarter, 0:32

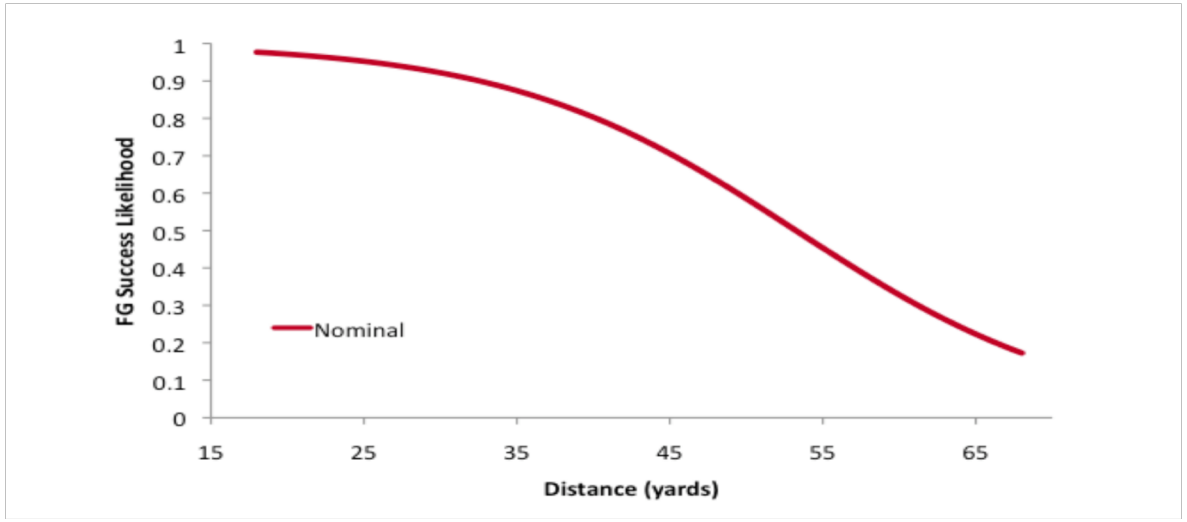
Analytics in the NFL

Based on the presentation:

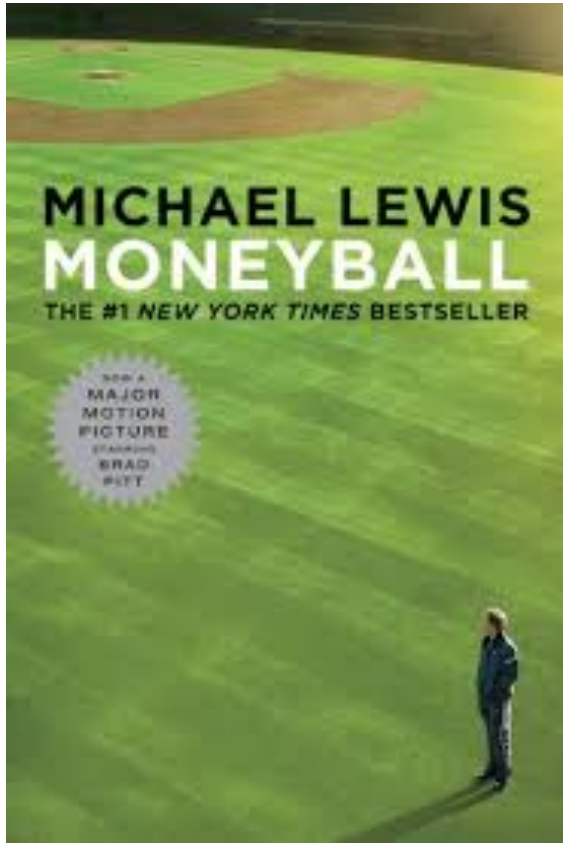
Going for Three: Predicting the Likelihood of Field Goal Success with Logistic Regression

Torin Clark, Aaron Johnson, Alexander Stimpson
Massachusetts Institute of Technology

	Variables	Coefficient	Significance
	Constant	$\beta_0 = 5.953$	p<0.0005
	Distance (yards)	$\beta_{dist} = -0.106$	p<0.0005
Environmental	Cold temperature (<50°F)	$\beta_{cold} = -0.341$	p<0.0005
Environmental	Field surface (artificial turf)	$\beta_{turf} = 0.299$	p<0.0005
Environmental	Altitude (≥ 4000ft)	$\beta_{alt} = 0.694$	p<0.0005
Environmental	Precipitation (rain, snow, etc.)	$\beta_{precip} = -0.280$	p=0.005
Environmental	Windy (≥ 10mph)	$\beta_{wind} = -0.140$	p=0.011
Environmental	Humid (≥ 60%)		p=0.844
Psychological / Situational	Postseason		p=0.196
Psychological / Situational	High situational pressure		p=0.539
Psychological / Situational	Away game		p=0.501
Psychological / Situational	"Icing the kicker" (TO before)		p=0.118

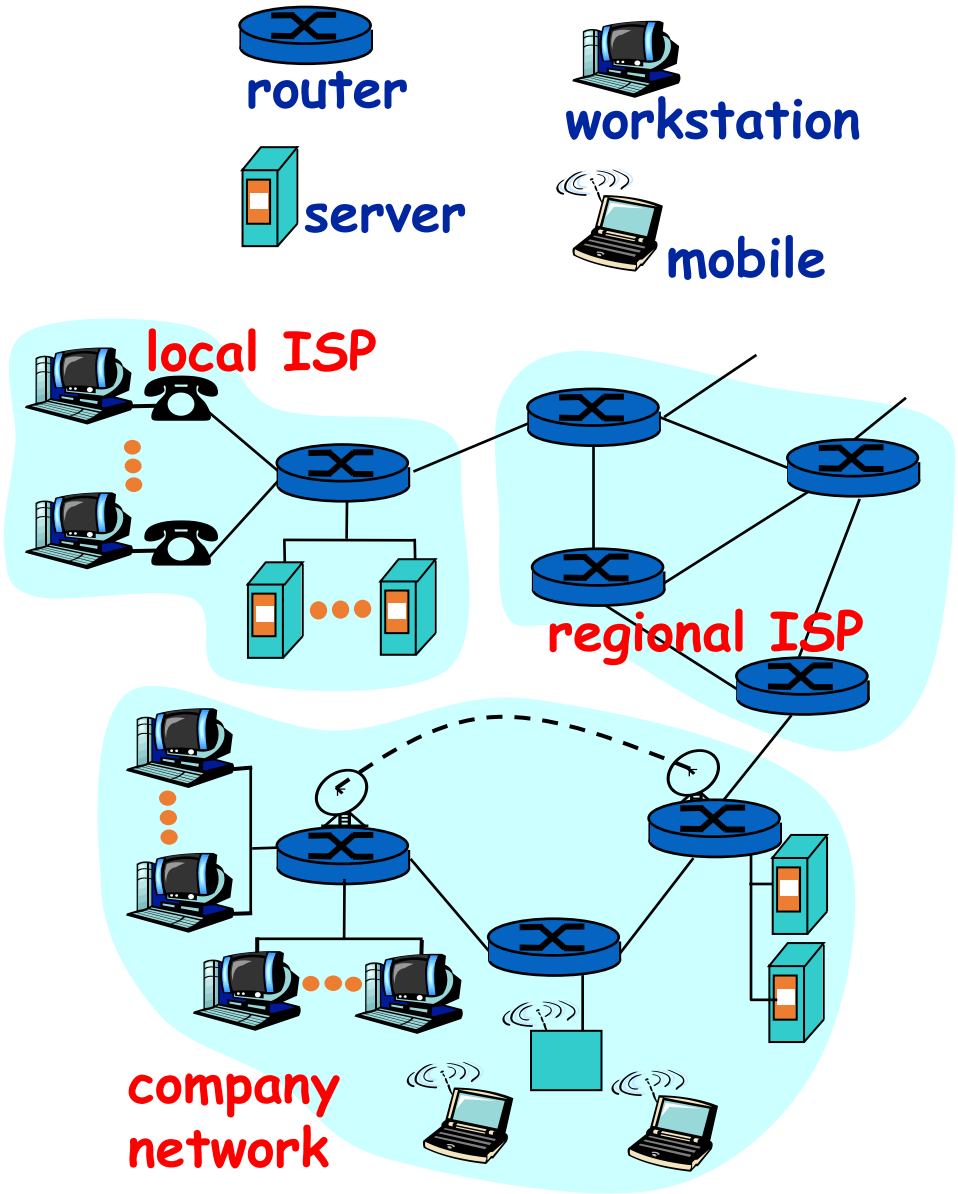


Moneyball


MICHAEL LEWIS
MONEYBALL
THE #1 NEW YORK TIMES BESTSELLER
NOW A MAJOR MOTION PICTURE STARRING BRAD PITT

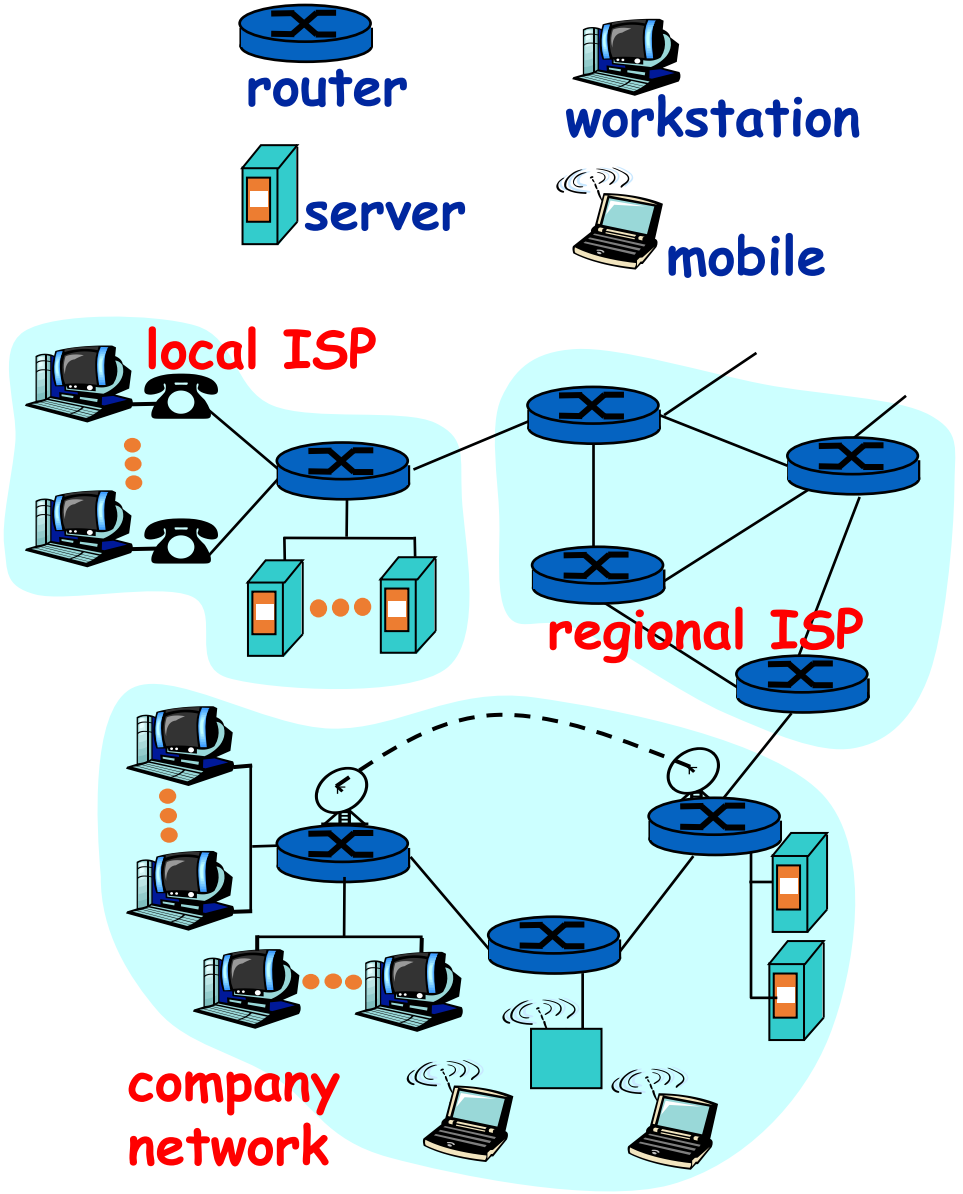

MONEYBALL

Internet data and resources



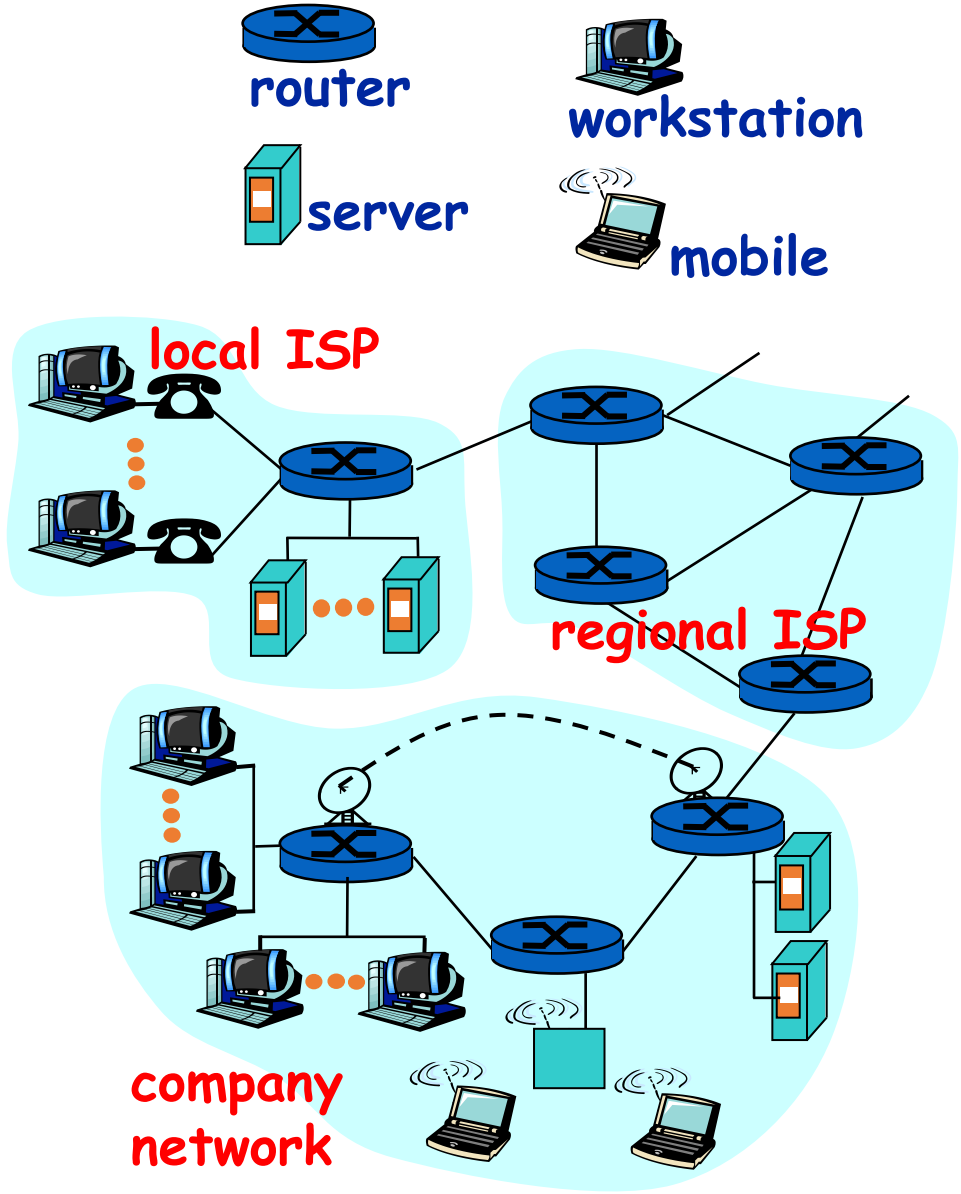
The Internet

- **Internet**: loosely hierarchical *network of networks*
  - Physical Components: Hosts, Routers, Communication links
  - Protocols: for exchanging messages
    - *TCP, IP, HTTP, FTP, PPP*



The Internet

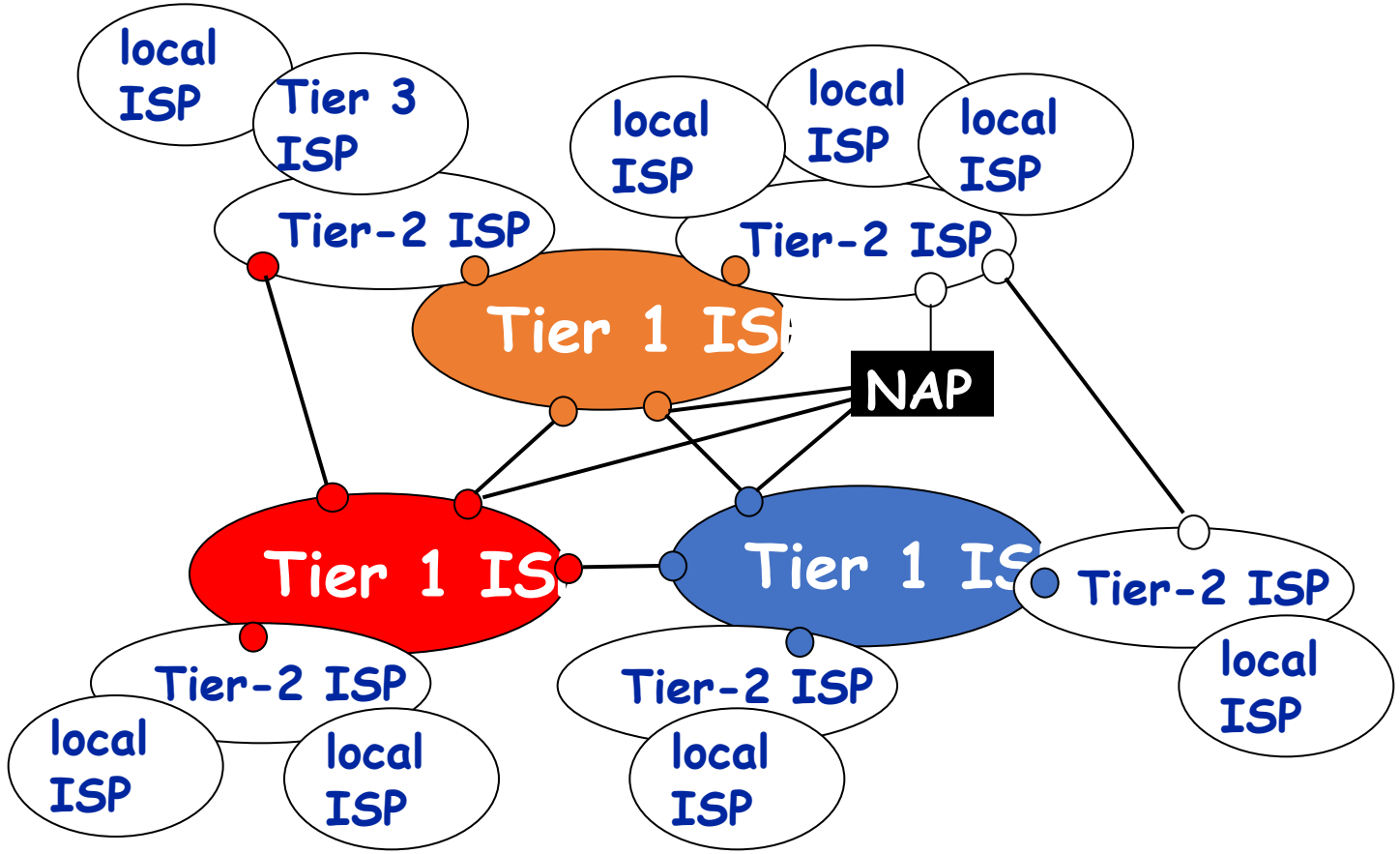
- Communication links:
  - Wifi, fiber, copper, radio, satellite
- Routers: forwarding packets (chunks of data)
- Hosts: connected computing devices executing network applications



The Internet

Hierarchical Architecture

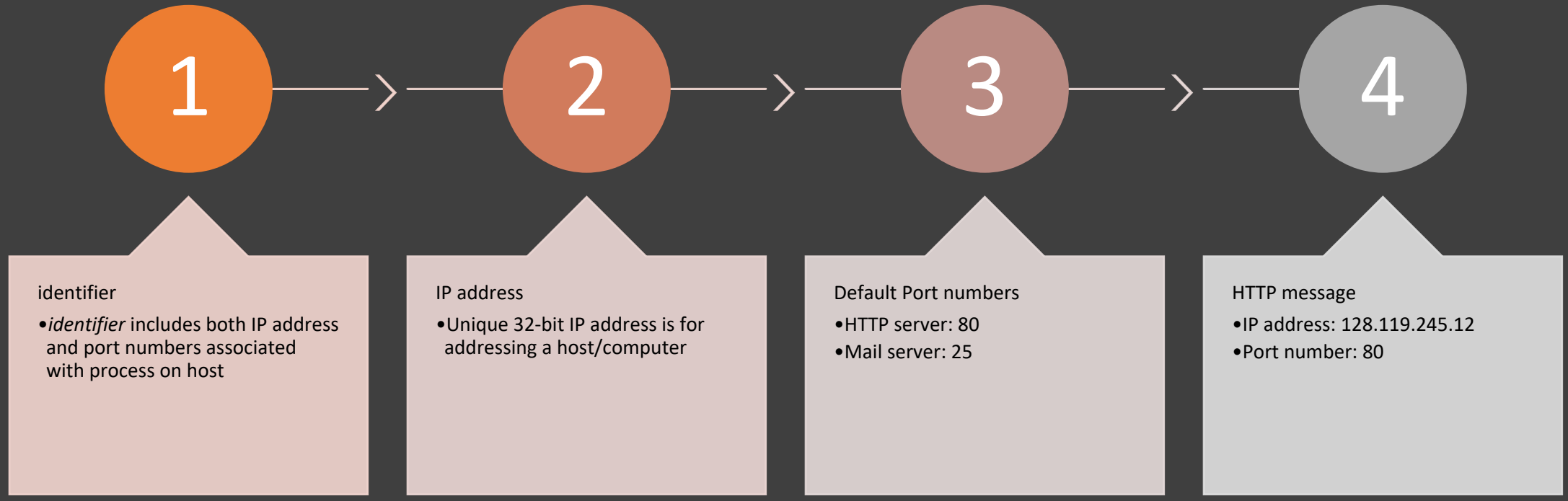
- Tier-1: ISPs like Verizon, AT&T
  - national/international coverage
- Tier-2: ISPs smaller/regional ISPs
- Tier-3 ISPs and local ISPs



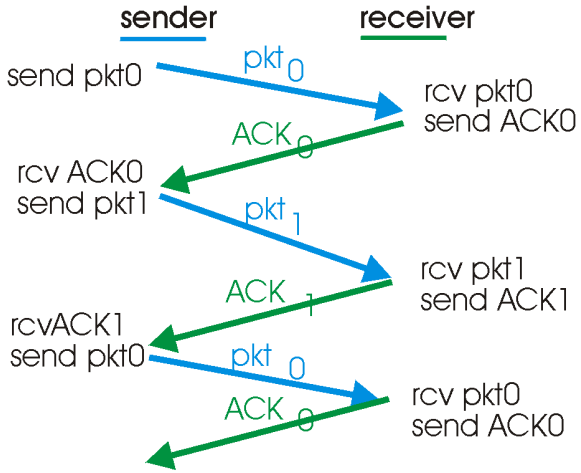
TCP/IP

Application:	• Supporting network applications • FTP, SMTP, HTTP
Transport:	• Host-host data transfer • TCP, UDP
network	• Routing of datagrams from source to destination
link	• Data transfer between neighboring networks • PPP, Ethernet
physical	• Bit transfer line or wireless

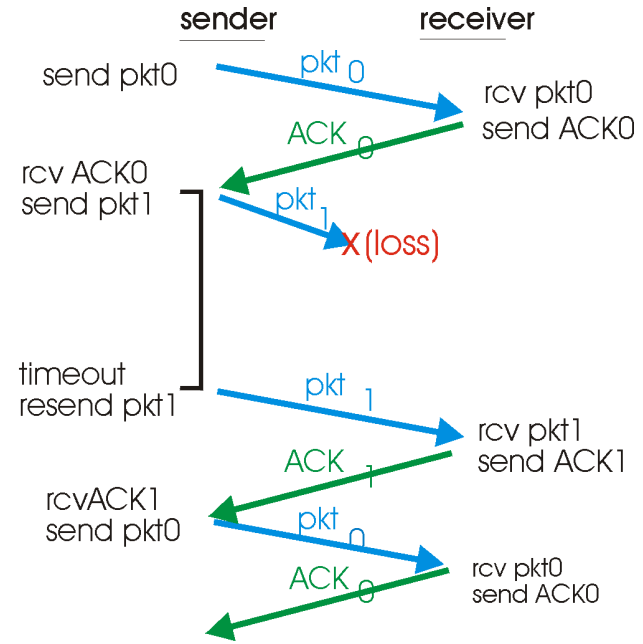
TCP/IP


1
2
3
4
identifier
•*identifier* includes both IP address and port numbers associated with process on host
IP address
•Unique 32-bit IP address is for addressing a host/computer
Default Port numbers
•HTTP server: 80
•Mail server: 25
HTTP message
•IP address: 128.119.245.12
•Port number: 80

TCP/IP



(a) operation with no loss



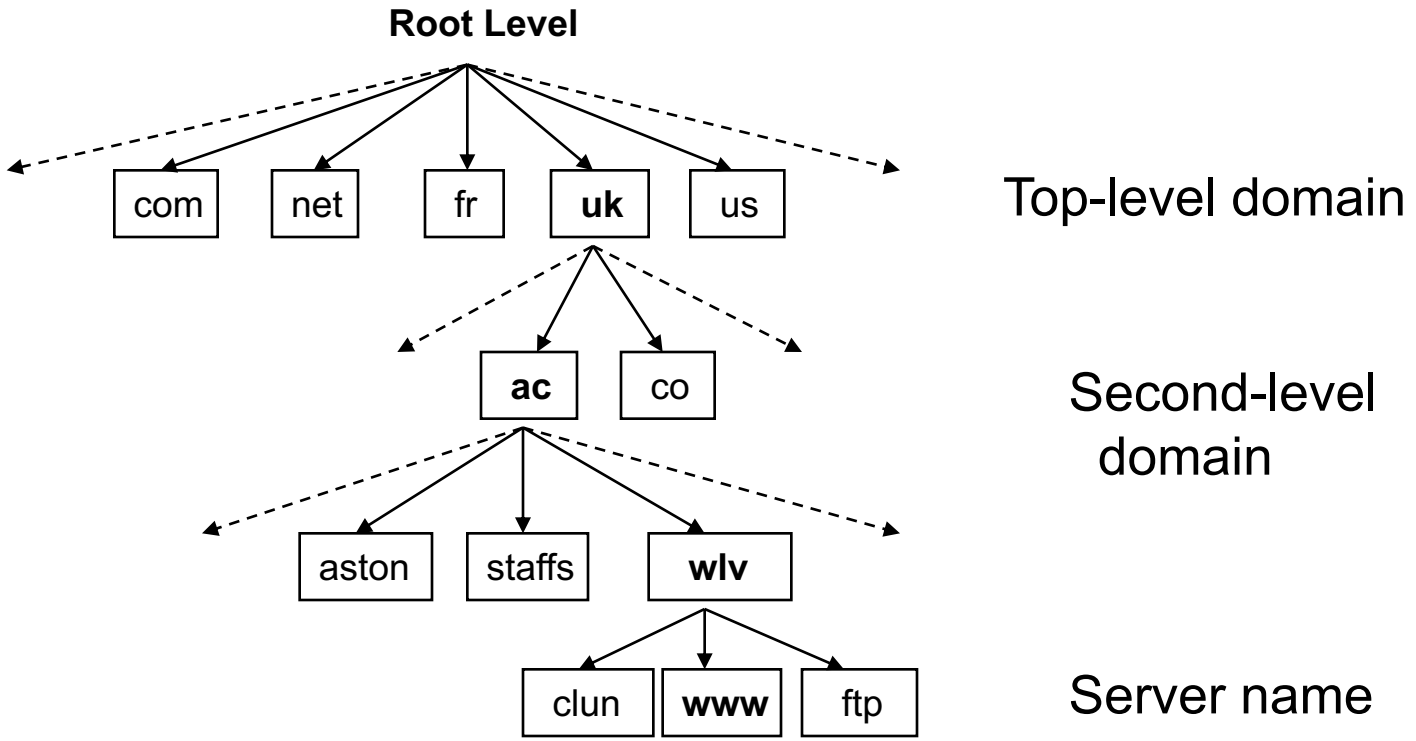
(b) lost packet

Domain Name System (DNS)

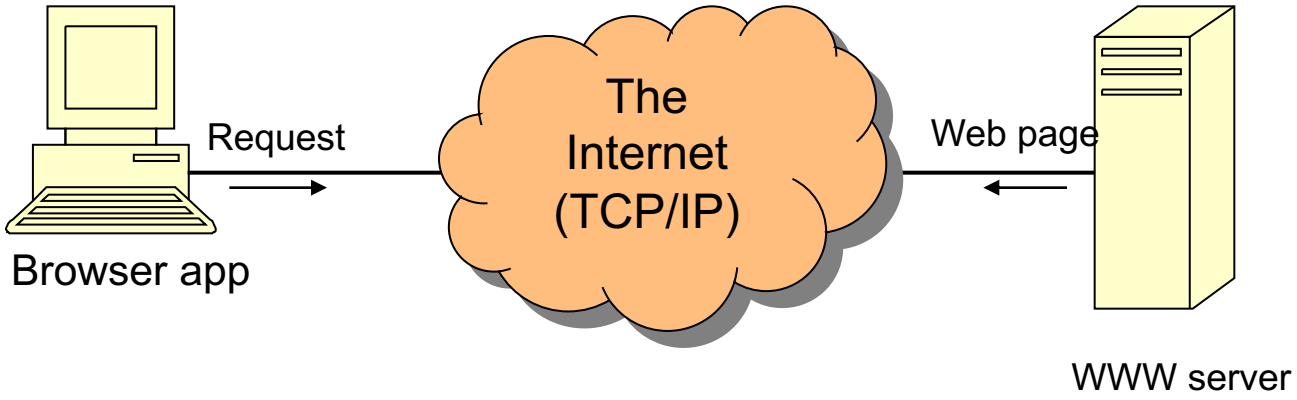
- IP addresses are used to identify hosts on a TCP/IP network
- Example: 134.220.1.9
- Numbers are not ‘friendly’ – people prefer names
- DNS is a protocol used to map IP addresses to textual names
- e.g. www.wlv.ac.uk maps to 134.220.1.9

Domain Name System (DNS)

DNS names have a hierarchical structure
Example: www.wlv.ac.uk



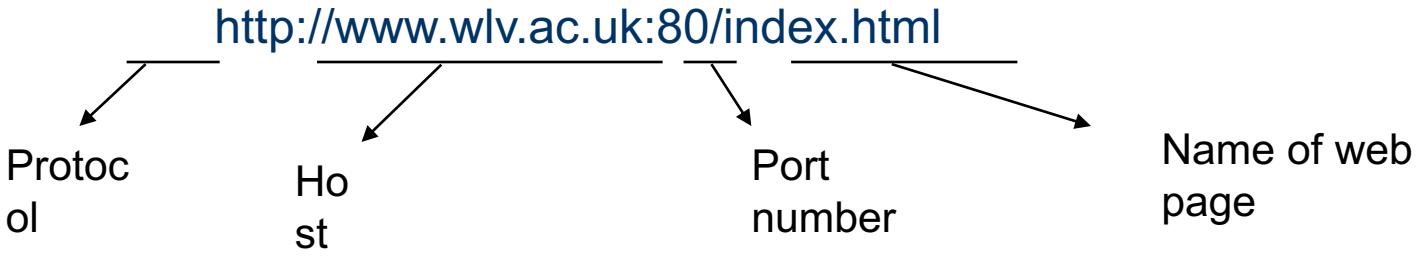
Domain Name System (DNS)



- HTTP is the protocol used to access resources on the World Wide Web
- A browser application is used to send a request to the WWW server for a resource, e.g. a web page, graphics file, audio file, etc.
- The server responds by sending the resource (a file) to the client and closing the connection

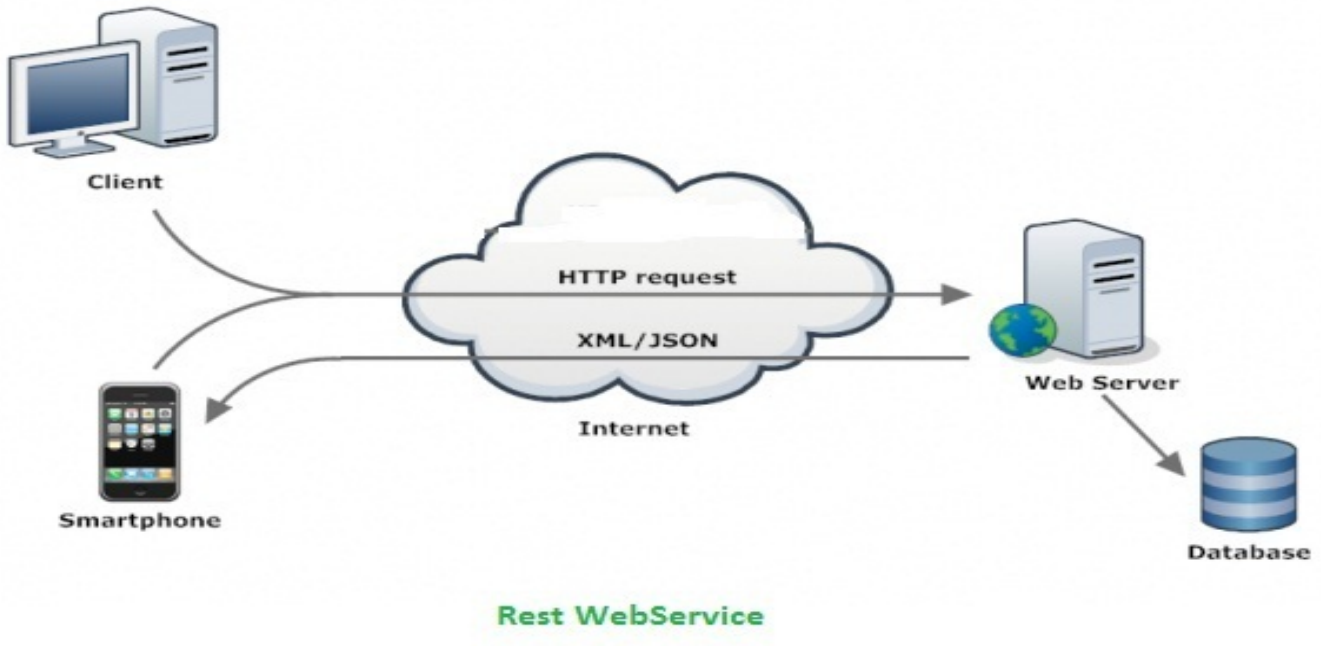
Uniform Resource Locator (URL)

- URL is the standard for specifying the whereabouts of a resource (such as a web page) on the Internet
- A URL has four parts:



  - The protocol used to retrieve the resource
  - The host where the resource is held
  - The port number of the server process on the host
  - The name of the resource file

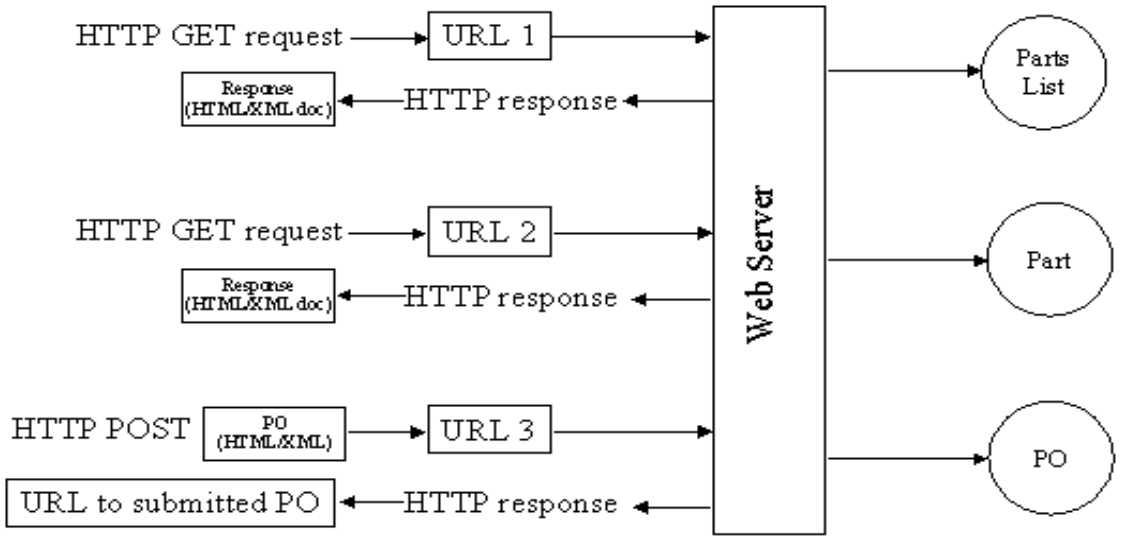
REST Architecture



Representational State Transfer [REST]

- An **architecture style** of networked systems
- Guiding framework for Web protocol standards
- Underlying Architectural model of the world wide web

REST Architecture



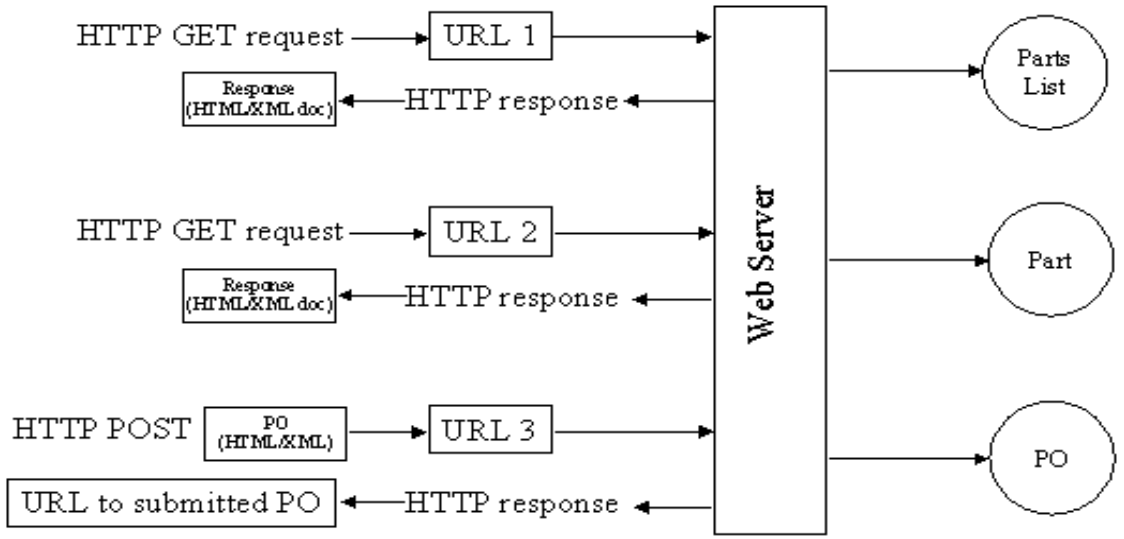
The web service makes available a URL to a parts list resource

Client uses : http://www.parts-depot.com/parts

Document Client receives :

```
<?xml version="1.0"?>
<p:Parts xmlns:p="http://www.parts-depot.com" xmlns:xlink="http://www.w3.org/1999/xlink">
    <Part id="00345" xlink:href="http://www.parts-depot.com/parts/00345"/>
    <Part id="00346" xlink:href="http://www.parts-depot.com/parts/00346"/>
    <Part id="00347" xlink:href="http://www.parts-depot.com/parts/00347"/>
    <Part id="00348" xlink:href="http://www.parts-depot.com/parts/00348"/>
</p:Parts>
```

REST Architecture



The web service makes available a URL to each part resource.

Client uses : http://www.parts-depot.com/parts**/00345**

Document Client receives :

```
<?xml version="1.0"?>
<p:Part xmlns:p="http://www.parts-depot.com" xmlns:xlink="http://www.w3.org/1999/xlink">
    <Part-ID>00345</Part-ID>
    <Name>Widget-A</Name>
    <Description>This part is used within the frap assembly</Description>
    <Specification xlink:href="http://www.parts-depot.com/parts/00345/specification"/> <UnitCost
     currency="USD">0.10</UnitCost>
    <Quantity>10</Quantity>
</p:Part>
```

Using the Command line - *curl*

```
➜ data ls
2019.01.22    2019.01.24    2019.01.29    2019.01.29_3C
➜ data curl http://www.nfl.com/ajax/scorestrip\?season\=2018\&seasonType\=POST\&week\=20
<?xml version="1.0" encoding="UTF-8"?>
<ss><gms gd="0" w="20" y="2018" t="P"><g eid="2019012000" gsis="57831" d="Sun" t="3:05" q=
"FO" k="" h="NO" hnn="saints" hs="23" v="LA" vnn="rams" vs="26" p="" rz="" ga="" gt="CON"/
><g eid="2019012001" gsis="57832" d="Sun" t="6:40" q="FO" k="" h="KC" hnn="chiefs" hs="31"
➜ dat➜ ➜ dat➜ dat➜ data
➜ data
```

curl http://www.nfl.com/ajax/scorestrip\?season\=2018\&seasonType\=POST\&week\=20
<?xml version="1.0" encoding="UTF-8"?>

Python urllib library

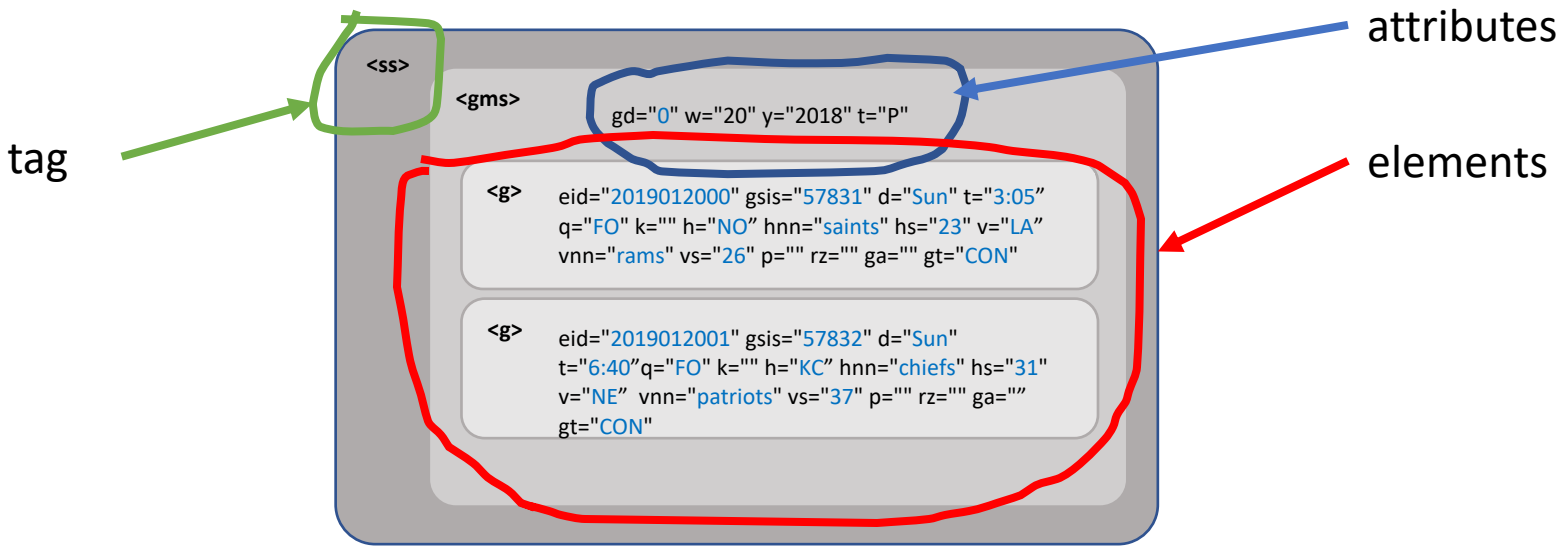
In[]:

```
from urllib.request import urlopen
query1='http://www.nfl.com/ajax/scorestrip?season=2018&seasonType=POST&week=19’
response = urlopen(query1)
reply_msg=response.read().decode("utf-8")
```

XML format

XML = Extensible Markup Language

- Tags
- Attributes
- Elements



XML format

```
<?xml version="1.0" encoding="UTF-8"?>
<ss>
   <gms gd="0" w="20" y="2018" t="P">
        <g eid="2019012000" gsis="57831" d="Sun" t="3:05”
           q="FO" k="" h="NO” hnn="saints" hs="23" v="LA”
            vnn="rams" vs="26" p="" rz="" ga="" gt="CON"/>
        <g eid="2019012001" gsis="57832" d="Sun" t="6:40”
           q="FO" k="" h="KC" hnn="chiefs" hs="31" v="NE”
           vnn="patriots" vs="37" p="" rz="" ga="" gt="CON"/>
    </gms>
</ss>
```

XML format

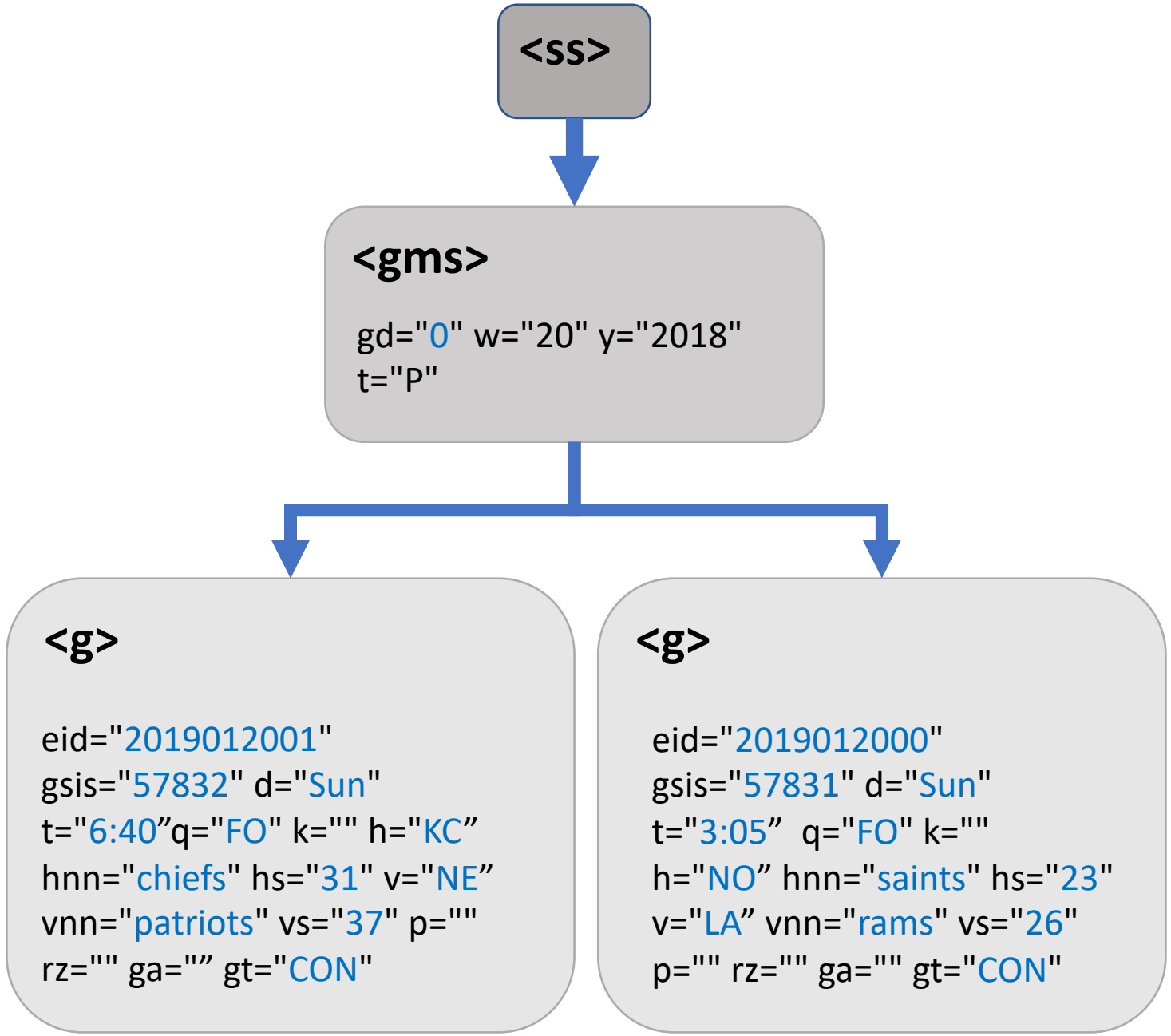
<ss>

<gms> gd="0" w="20" y="2018" t="P"

<g> eid="2019012000" gsis="57831" d="Sun" t="3:05”
q="FO" k="" h="NO” hnn="saints" hs="23" v="LA”
vnn="rams" vs="26" p="" rz="" ga="" gt="CON"

<g> eid="2019012001" gsis="57832" d="Sun"
t="6:40”q="FO" k="" h="KC” hnn="chiefs" hs="31"
v="NE” vnn="patriots" vs="37" p="" rz="" ga=""”
gt="CON"

XML format


<ss>
<gms>
gd="0" w="20" y="2018"
t="P"
<g>
eid="2019012001"
gsis="57832" d="Sun"
t="6:40”q="FO" k="" h="KC”
hnn="chiefs" hs="31" v="NE”
vnn="patriots" vs="37" p=""
rz="" ga="” gt="CON"
<g>
eid="2019012000"
gsis="57831" d="Sun"
t="3:05” q="FO" k=""
h="NO” hnn="saints" hs="23"
v="LA” vnn="rams" vs="26"
p="" rz="" ga="" gt="CON"

ElementTree

In[]:

```
from urllib.request import urlopen
query1='http://www.nfl.com/ajax/scorestrip?season=2018&seasonType=POST&week=19'
response = urlopen(query1)
xml_doc=response.read().decode("utf-8")
```

In[]:

```
import xml.etree.ElementTree as ET
root = ET.fromstring(xml_doc)
node_ss=root.getchildren()
print(len(node_ss))
node_gms=node_ss[0].getchildren()
print(len(node_gms))
for node_g in node_gms:
    print(node_g.attrib)
```

ElementTree

In[]:

```
root = ET.fromstring(xml_doc)
node_ss=root.getchildren()
print(len(node_ss))
node_gms=node_ss[0].getchildren()
print(len(node_gms))
for node_g in node_gms:
    print(node_g.attrib)
```

1
4

{'eid': '2019011200', 'gsis': '57827', 'd': 'Sat', 't': '4:35', 'q': 'F', 'k': '', 'h': 'KC', 'hnn': 'chiefs', 'hs': '31', 'v': 'IND', 'vnn': 'colts', 'vs': '13', 'p': '', 'rz': '', 'ga': '', 'gt': 'DIV'} {'eid': '2019011201', 'gsis': '57828', 'd': 'Sat', 't': '8:15', 'q': 'F', 'k': '', 'h': 'LA', 'hnn': 'rams', 'hs': '30', 'v': 'DAL', 'vnn': 'cowboys', 'vs': '22', 'p': '', 'rz': '', 'ga': '', 'gt': 'DIV'}

{'eid': '2019011300', 'gsis': '57829', 'd': 'Sun', 't': '1:05', 'q': 'F', 'k': '', 'h': 'NE', 'hnn': 'patriots', 'hs': '41', 'v': 'LAC', 'vnn': 'chargers', 'vs': '28', 'p': '', 'rz': '', 'ga': '', 'gt': 'DIV'}
{'eid': '2019011301', 'gsis': '57830', 'd': 'Sun', 't': '4:40', 'q': 'F', 'k': '', 'h': 'NO', 'hnn': 'saints', 'hs': '20', 'v': 'PHI', 'vnn': 'eagles', 'vs': '14', 'p': '', 'rz': '', 'ga': '', 'gt': 'DIV'}

JSON format

JSON – Java Script Object Notation

- A simple text-oriented format for data exchange between a browser and a server
- Inspired by JavaScript object literal syntax, but nowadays used well beyond the JavaScript world
- Became one of the most popular data exchange formats in the last years

JSON format

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<ss>
   <gms gd="0" w="20" y="2018" t="P">
        <g eid="2019012000" gsis="57831/>
        <g eid="2019012001" gsis="57832/>
    </gms>
</ss>
```

JSON

```
{
   {
     “gd”="0"
      “w”="20"
     {
       “eid”="2019012000”
       “gsis”="57831”
      }
      {
        “eid”="2019012001”
        “gsis”="57832”
       }
     }
}
```

JSON

In[]:

```
from urllib.request import urlopen
query2='http://www.nfl.com/liveupdate/game-center/2019012001/2019012001_gtd.json'.format(eid)
response = urlopen(query2)
json_string=response.read().decode("utf-8")
print(json_string)
```

{"2019012001":{"home":{"score":{"1":0,"2":0,"3":7,"4":24,"5":0,"T":31},"abbr":"KC","to":3,"stats":{"passing":{"00-0033873":{"name":"P.Mahomes","att":31,"cmp":16,"yds":295,"tds":3,"ints":0,"twopta":0,"twoptm":0}},"rushing":{"00-0030874":{"name":"Dam. Williams","att":10,"yds":30,"tds":1,"lng":10,"lngtd":2,"twopta":0,"twoptm":0},"00-0033873":{"name":"P.Mahomes","att":2,"yds":11,"tds":0,"lng":9 …

JSON

In[]:

```
parsed_json = json.loads(json_doc)
print(parsed_json)
```

{'2019012001': {'home': {'score': {'1': 0, '2': 0, '3': 7, '4': 24, '5': 0, 'T': 31}, 'abbr': 'KC', 'to': 3, 'stats': {'passing': {'00-0033873': {'name': 'P.Mahomes', 'att': 31, 'cmp': 16, 'yds': 295, 'tds': 3, 'ints': 0, 'twopta': 0, 'twoptm': 0}}, 'rushing': {'00-0030874': {'name': 'Dam. Williams', 'att': 10, 'yds': 30, 'tds': 1, 'lng': 10, 'lngtd': 2, 'twopta': 0, 'twoptm': 0}, '00-0033873': {'name': 'P.Mahomes', 'att': 2, 'yds': 11, 'tds': 0, 'lng': 9, 'lngtd': 0, 'twopta': 0, 'twoptm': 0}}, 'receiving' …

JSON

In[]:

```
parsed_json['stats']
```

```
{'2015091000':
    {
     'passing':
        {
         '00-0019596':
            {
             'name': 'T.Brady',
             'att': 32,
             'cmp': 25,
             'yds': 288,
             'tds': 4,
             'ints': 0,
             'twopta': 0,
             'twoptm': 0}},
     'rushing':
            {
             '00-0028087':
               {'name': 'D.Lewis',
                'att': 15,
                'yds': 69
                ...
```

JSON

In[]:

```
parsed_json['stats']['2015091000']['rushing']
```

```
{'00-0028087':
        {'name': 'D.Lewis',
                 'att': 15,
                 'yds': 69
                 …
```